How to begin working on any Dataset in R Studio

#getTidy 20Oct19 – Parisi

Good for when you wanna jump in but realize you don't know all the simple commands to start…

1.) Load packages
    a. Library(foreign) should help bring in the read commands
    b. Library(dplyr) will give you access to the glimpse() command, some joining stuff, yeah it's good to have
    c. read.fileformat("Filename.ext")
    d. getwd() and setwd("C:/…") will get your working directory (where R knows to look) and set wd to a new location aka where your files are store that you want to read!
2.) Explore
    a. Feel up the data that you've got, see how crappy it is, get an idea for where data is at and where u goin
    b. How big is it? dim(), nrow(), ncol(), summary(), str(), glimpse(), colnames()
    c. What type is it? class()
3.) Bring in other data, or create a new easier matrix to work with… (advised)
    a. cbind(data1,data2)
    b. rename the default heading with colnames(Data) <- c("colname1","colname2",…)
4.) Look for and get rid of NA's
    a. Summary(data) should show you all the values that appear for a given variable/column
        i. You may find weird values like 'dk', 'NaN', etc.
        ii. Change them all to NA which is what R likes
            1. StudentCollege$college[StudentCollege$college == 'dk'] <- NA
            2. Note the $ is used to pick out a column/variable within the dataset
    b. To find NA's you can use the any(complete.cases(data))
        i. Complete.cases looks to see if an observation has values for every variable
            1. Returns TRUE if it is a complete case and FALSE otherwise
    c. If your NA's are truly garbage observations and you want to throw them out
        i. StudentCollege<-StudentCollege[complete.cases(StudentCollege),] discards all rows w/ >=1 NA
5.) Change the Type of Data
    a. Now you've got it all together in one big mosh, you gotta let R better *understand* the data you have
    b. Class() to check the current class of variables and such
    c. as.numeric() for continuous numerical values, as.character() for text strings fits categorical, as.factor() for the ordinal, as.integer() for discrete numerical, these change the data type
    d. Types of variables in general
        i. numeric/quantitative
            1. discrete → 5 people, 8 people in a room (finite values, can't have 5.67 people)
            2. continuous → 1.15, 1.67, 1.88, can be infinite values
        ii. categorical/qualitative
            1. categorical → no intrinsic order or rank, "pumpkins, cherries, apples"
            2. ordinal → inherent order like "never, often, all the time man!"
    e. with your big set, currently it's a matrix, use as_data_frame(data) #POWERFUL
        i. now try glimpse(), should be thick
        ii. you can also call out data$column1 and use a BUNCH more functions like joins! Woot!
6.) Joins – got something else you wanna bring in? Cool.
    a. Left_join(maindata,datatoadd,variablestoaddon = "variablename")
        i. Columns from datatoadd will be added to maindata making those observations more fruitful, but only for the observations in maindata that have the same variablename as datatoadd
            1. Thus the rows/observations will stay the same, but the columns will increase with the unique columns that were a part of datatoadd
7.) Export
    a. write.table(Dataname, file="Filename.csv", sep=",") writes as a comma separated values file exported into the working directory, then you can open this in excel